Version 2 January 2022. **N2X3**

Instruction sets of the English Electric KDN2, KDF6 and KDF7 computers.

Descriptions of the instruction sets are presented in sequence, with each of the three computers appearing on a new page.

The KDN2 Instruction Set.

The KDN2 has a single-address instruction format. Of the 18-bit instruction, six bits are employed for the op code (giving a theoretical maximum of 64 functions) and 12 bits are used to specify the operand address in the core store. The ferrite core primary store was expandable in increments of 512 18-bit words up to a maximum of 4,096 18-bit words.

The usual add, subtract, multiply and divide operations are provided. There are both arithmetical and logical shift instructions. Three types of control transfer instructions are provided: unconditional jumps, jump if zero, jump depending upon the sign of the operand.

At the time of writing, English Electric documentation describing the full instruction set has not been discovered.

Here are some sample instruction times:

Add/subtract and logical ops	175 microsecs. (18-bit aithmetic)
Jump	42 microsec.
Instruction modify:	168 microsec.
Multiplication:	average: 2.75 millisecs., maximum 5 millisecs.
Divide	5 millisec.
Input/output:	91 microsecs. per 6-bit character.

The English Electric DEUCE Alphacode was also made available for the KDN2 computer.

The KDF 6 Instruction Set.

The description below is based on the English Electric brochure *KDF 6 Programming,* as transcribed by Andrew Herbert in his KDF 6 article in *Resurrection,* number 89, Spring 2020.

The KDF 6 had a main store of 24K 18-bit words and four registers named **A**, **B**, **C**, **D**. Registers **A** and **B** can be concatenated as an extended register called **X**. A word could be interpreted as an integer, or as three six-bit characters ("triads"). In machine code an instruction was represented by two octal digits (six bits – allowing for 32 instruction codes) for the function code and four octal digits addressing an operand in main store (12 bits – allowing for 4096 locations).

Octal	*Mnemonic*	*Interpretation; comments.*
00,01	**CALL, SEND**	Input / Output
02,04, 06, 03,05, 07	**MOP B, MOP C, MOP D, MOM B, MOM C, MOM D**	The contents of the designated register are added to (**MOP**) or subtracted from (**MOM**) the operand and executed as an instruction
10,12, 14,16, 11,13, 15,17	**SET A, SET B, SET C, SET D, STR A, STR B, STR C, STR D**	**SET** fetches the operand to the register **STR** copies the register into the operand address
20,22, 24,26, 21,23, 25,27	**ADD A, ADD B, ADD C, ADD D, SUB A, SUB B, SUB C, SUB D**	Add the operand to the register / subtract the operand from the register
30,32, 34,36, 31,33,	**NEV A, NEV B, NEV C, NEV D, AND A, AND B, AND C, AND D**	bitwise not equivalent of register and operand to register, bitwise logical and of register and operand to register
40,42, 44,46, 41,43, 45,47	**SHL A, SHL B, SHL C, SHL D, SHR A, SHR B, SHR C, SHR D**	shift register left/right by number of places specified in operand field. **SHR A** and **SHR D** are arithmetic shifts, **SHR B** and **SHR C** are logical shifts
50,52, 54,56, 51,53, 55,57	**JNZ A, JNZ B, JNZ C, JNZ D, JNN A, JNN B, JNN C, JNN D**	Jump to operand address if register is non-zero, non-negative and continue
60,62, 64,66, 61,63, 65,67	**INC A, INC B, INC C, INC D, DEC A, DEC B, DEC C, DEC D**	Increment / decrement operand in store and copy result to register
70 71	**SHL X, SHR X**	shift contents of extended register **X** up or down by operand field places. **SHR X** is arithmetic

72 **73**	**ADD X,** **SUB X**	Add/subtract unsigned contents of operand field to/from extended register **X**
74	**DVDE**	Divide **X** by **C** putting quotient in **B** and remainder in **A**. Operand field specifies number of places of division required
75	**MULT**	Multiply **C** by **B** putting result in **X**. The operation field the number of cycles of multiplication (normally 18). If **A** is non-zero before the operation its contents are added into the result
76	**JUMP**	Jump to the operand address and continue
77	**HOLD**	Jump to the operand address and halt.

The instruction times are relatively slow:

MOM, MOP	168μs + instruction obeyed
SET	175μs
STR	196μs
ADD, SUB	175μs
ADD X, SUB X	301μs
MULT	2750μs (average)
DVDE	5222μs (average)
NEV, AND	175μs
SHL, SHR	35+7Y μs (Y=number of places to shift)
JNZ, JNN, JUMP	42μ
JNZ, JNN, JUMP	196μs

Function codes (op codes) can be written in octal, or as the mnemonics given above.

The operand field can be:

- Absolute, i.e., a decimal or octal number e.g., **ADD ,100**
- Relative to the address of the instruction e.g., **JUMP *-8**
- Symbolic: relative to a label, e.g., **JUMP R4/-2**

The brochure *KDF 6 Programming* is not explicit about addressing, although the term “core module” appears from time to time in the documents but is undefined. It could be that operand addressing is absolute in the range 0-4095, relying on **MOP/MOM** instructions to access higher regions of store, or possibly relative to the module in which the program is executing.

Input/output to slow devices is directly through registers, character by character. For fast devices, block transfers can be set up from buffers in the second core module and program execution will continue in parallel with data transfer. Only one block transfer may be in process at the time and there are no facilities for multi-programming (i.e., interrupts). The block transfer facility is provided by library subroutines and the KDF 6 User Code provides special mnemonics to set up calls to these routines.

Somewhat mysteriously, the brochure says: “Data areas: any triad in a data area may be addressed by a symbolic address followed consisting of a minuscule followed by a decimal number. Data areas must be defined by directives at the beginning of the program”.

Labels can be:

1. Reference labels (in the main program)
2. Subroutine reference labels (for references to subroutines)
3. Entry point labels (for defining entrances to subroutines from the main program)
4. V-Labels (for internal references within subroutines
5. Symbolic labels (these enable subsequent instructions and groups of constants to be treated as data areas).
6.

Unfortunately, there are no examples in the brochure to reveal the syntax of each label type.

The KDN7 Instruction Set.

At the time of writing, no English Electric technical manuals have come to light that give details of the KDF7's internal register layout and instruction repertoire. Until fresh evidence comes to light, it might be assumed that the KDF7 was an upgraded version of the KDN2 computer.